A PROJECT SUPPORT ENVIRONMENT REFERENCE MODEL

Alan Brown, David Carney, and Peter Feiler
Software Engineering Institute
Carnegie Mellon University
Pittsburgh
PA 15213

Patricia Oberndorf
NAWC/AD
Code 7031
P.O. Box 5152
Warminster, PA 18974

Marvin Zelkowitz
Dept. of Computer Science
University of Maryland
College Park
MD 20742

Abstract

The Navy's Next Generation Computer Resources (NGCR) program set up a Project Support Environment Standards Working Group (PSESWG) to help in the task of establishing interface standards that will allow the U.S. Navy to more easily and effectively assemble software-intensive Project Support Environments (PSEs) from commercial sources. A major focus of PSESWG is the development of a service-based reference model that will provide the context for categorizing and relating existing standards and the identification of interface areas that may benefit from future standardization. This paper presents a report on this reference model.

1. INTRODUCTION

From its inception the Ada effort saw the critical role played by tools that support application development (i.e., the notion of a programming support environment). The seminal work described in the Stoneman report [Stoneman 80] emphasized not only the role of tools for language support such as editors and compilers, it also went much further than that in recognizing the need for:

- an extended set of tools that allow smooth transition between phases of the life cycle.
- well defined interfaces that support interoperability and portability of tools across environment implementations.

While Stoneman focused on *programming* support, the same model has been extended to *project* support where the application domain is the wider context of software development, and is not limited to systems written in Ada. This overall support environment (hardware, software, methods and techniques, people) can have a significant effect on the quality of the computer system developed, and the ease (or otherwise) with which it can be maintained. Having well defined interfaces for tools appears to be a critical factor in achieving the goal of a "plug-and-play" approach to environment assembly.

Significant work since the Stoneman report has attempted to refine the notion of a project support environment (PSE), where the term has been expanded to refer to a computer-based system used in developing, maintaining, and enhancing a computer systems. PSEs are currently being studied and used by many organizations in government and industry. Many organizations are seeking ways more easily to develop PSEs that are specific to particular projects or individuals. The goal of most of these efforts is finding a strategy that permits a PSE to be constructed from commercial off-the-shelf (COTS) tools in a flexible and maintainable way. Unfortunately, while sound in concept, this approach suffers from the current instability and fragmentation of the COTS tool market, with the result that assembling a PSE from a collection of COTS tools is a very complex undertaking. Not only are there many different COTS tools to choose from, there are many tools offering similar functionality, new tools being announced by vendors on a frequent basis, and no established means to use multiple tools within a single PSE. While often talked about, the notion of "plug-and-play" compatibility in COTS tools remains a long way from reality.

Effective use of a PSE requires that tools are integrated according to several criteria, or dimensions, to provide consistent interfaces with other related tools. The three dimensions most often mentioned are:

1. Control — how tools sequence their execution among one another.
2. Data — how tools pass or share information that each of them needs.
3. Presentation — how different tools present information to end users in a consistent manner.

While we understand the needs for such integration criteria, we do not yet have an agreed upon set of mechanisms for supporting those needs.

As a means to resolve this dilemma, many working groups are investigating how standardized interfaces (actually sets of complementary interfaces) can provide the necessary mechanism for tool integration. The argument used is that a set of interfaces that are publicly defined and agreed can act as the basis for interoperability and portability of support tools. As a first step, it is necessary to isolate the specific areas for which interface standards are needed. It is our belief that a suitable abstract model of a PSE, usually called a *reference model*, is a prerequisite for accomplishing this.

1.1. Background to the Reference Model

As a developer, acquirer, supporter, and user of numerous large, software-intensive computer applications, the U.S. Navy recognizes the importance of improving its approach to all aspects of computer use. The Next Generation Computer Resources (NGCR) is a U.S. Navy program designed to establish industry-based interface standards in a number of areas important to mission-critical computer resources (MCCR). Recognizing the current state of the practice in the area of support environments, the Navy decided as part of the overall NGCR program to focus one of its teams on the area of PSEs[1]. It consists of participants from industry and academia, as well as a variety of government services and agencies. The PSE Standards Working Group (PSESWG) is selecting interface standards that will help the Navy in moving toward the goal of being able to assemble a PSE from COTS tools in a well-defined way.

The PSESWG was initiated in February 1991 with a charter to establish an industry-based set of environment interface standards. These standards, and the environments that conform to them, must be suitable for supporting engineering and management through the entire life-cycle of computer-based applications systems in the 1990s and beyond. Two related tasks were initiated as a starting point to achieve the PSESWG charter:

1. The development of a PSE reference model. Due to the complexity and lack of agreed terminology and concepts in this area, it was decided to develop a model based on the characterization of the facilities expected of a populated PSE. These facilities include both the support services and the tools that provide capabilities to the end-user.
2. A cataloging of available technology and standards. As there are many existing standards and products already in use, there was a need to begin the task of identifying and categorizing those that might be relevant to the PSESWG.

There is a close relationship between these two tasks in that the aim is to validate the reference model by applying knowledge of existing products and standards, and to use the categories and partitions of the reference model to help in structuring the available technology catalog.

The PSE reference model is the starting point for selecting interface standards. It will provide the context for classifying existing products and standards, establish a common terminology and set of concepts for PSESWG (and perhaps the broader PSE community), and identify where further standards efforts may need to be directed. In developing this reference model we examined a large number of existing PSE efforts. None of them individually had the breadth, nor had the approach necessary to achieve our aim of providing a vehicle for identifying interfaces as potential candidates for standardization in a PSE. Hence, our model synthesizes aspects of many of them, including the Software Technology for Adaptable, Reliable Systems (STARS) program, the National Institute of Standards and Technology (NIST) Integrated Software Engineering Environment (ISEE) working group, the European Computer Manufacturers Association (ECMA) Technical Committee 33 task group on the SEE reference model, the Ada Joint Program Office Evaluation and Validation Team, the Air Force Software Life Cycle Support Environment (SLCSE) project, Honeywell's Engineering Information System (EIS) program, TRW's Conceptual Environment Architecture Reference Model (CEARM) effort, and the standardization committees within IEEE and ANSI for POSIX and for CASE Tool Integration Models (CTIM). Many valuable aspects of these efforts have been considered in the course of our work.

1.2. Overview of this Paper

Section two describes the basis of the reference model, together with a description of the key terms and concepts that provide a basis for this work.

Section three describes the services defined in the model itself.

The paper is concluded in section four which summarizes the main issues raised and describes ongoing activities related both to the reference model and to other related research activities.

1. In addition to the PSESWG, other working groups are concentrating on network, backplane bus, operating systems, database, and graphics interface standards.

2. DESCRIPTION OF THE MODEL

The reference model is a conceptual description of the functionality that may be provided by a project support environment.[2] This description is general and is bounded neither by a particular application domain, by a particular programming language, nor by any specific life-cycle paradigm for a development project. This is in contrast to an actual implemented environment that is constructed of particular components (i.e., software and hardware) supporting one or more programming languages and that typically does reflect a chosen life cycle paradigm, at least implicitly.

The distinction between conceptual and actual is of fundamental importance. The conceptual viewpoint that governs this reference model provides an abstract description of the functionality that may be found in a PSE. An actual viewpoint would describe a particular realization of the conceptual view in terms of a PSE architecture with specific tools and standards. There is a mutually reflective relationship between the conceptual and the actual views, i.e., between this PSE reference model and existing environments: one may either consider the model to be abstracted from many environments, or may regard a particular environment as a realization of the model.

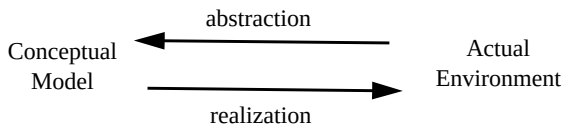

Figure 1: Conceptual Models vs. Actual Environments.

Figure 1 illustrates this distinction. The left-pointing arrow illustrates the activity of studying several existing environments to derive information for the model. The right-pointing arrow indicates that a particular environment could be a realization of the model. One benefit of this approach is that it provides a common means of describing environments (e.g., "In terms of their functionality, how is SLCSE[3] different from EAST[4]?"). Hence, the reference model does not directly represent an architectural approach to constructing a PSE — it provides a common basis for examining the functionality of different PSEs. Analysis of existing environments also plays another important role — supporting an ongoing validation of the model; it is a necessary attribute that the reference model provides an accurate reflection of technology that exists, even as the technology evolves over time.

2.1. Key Concepts and Terms

There are several key concepts and terms used in the reference model. This section provides an overview of them and their interrelationships. These key terms are indicated below by italics. It should be noted that some of these concepts are not amenable to simple definition, either because the term is broadly applicable, forcing description rather than definition, or because the term currently has conflicting meanings in the environments community. It is hoped that this section of the paper may help resolve some of this confusion.

An *Environment* is a collection of software and hardware[5] components; there is typically some degree of compatibility that renders these components harmonious. One can describe an environment using the contrasting viewpoints of conceptual vs. actual; or in a slightly different way, one can describe an environment in terms of the way it supports human activities.

When described from the conceptual point of view, an environment's capabilities are referred to as *Services*, which are abstract descriptions of the work done. Some of these services are of direct interest to an *End-user* (e.g., editing) while others comprise an underlying infrastructure, or *Framework*, comprised of relatively fixed capabilities that support user interfaces, processes, and objects (e.g., access control, process resource management).

When described from the opposite, or actual, view, i.e., when a realized environment is considered, the components that directly support an end-user are generally called *Tools* (e.g., Ada compilers, graphical design packages). Although no single definition for "tool" will suffice, that of the IEEE Glossary[6] is useful: a computer program used to help develop, test, analyze, or maintain another computer program or its documentation. As in the conceptual view, the components that comprise an actual infrastructure are referred to as the *Framework*. The same term, framework, is thus used in both a conceptual and an actual sense, and its precise meaning depends on the context.

Finally, when an Environment is considered from the vantage point of how it supports human activities, then either the environment will provide a *Service* to a human user, or a human user will perform some *Task* with the aid of the environment. For instance, one can speak of the *task* of

2. Although the term "environment" has not yet been fully defined, the reader is presumed to have some familiarity with the term as commonly used.

3. The Software Life Cycle Support Environment, a software development sponsored by the U.S. Air Force.

4. The Environment of Advanced Software Technology, a product developed by SFGL in France.

5. For the purposes of this document, the PSESWG concentrates on the software components of an environment.

6. *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std 610.12-1990.

testing software, or of using a software testing *service*. These different views of an environment result in subtle differences in the meanings of key terms. In particular, there is a slightly different meaning for service when it is contrasted with tool and when it is contrasted with task. In the first case, a tool is an actual realization of one or more conceptual services. While there is no strict correlation between tool and service (because one tool may realize many services, or one service may be realized by many tools), there are relatively straightforward correlations between tools' functionalities and service descriptions. In the second case, a task and a service provide complementary views of the same activity. For

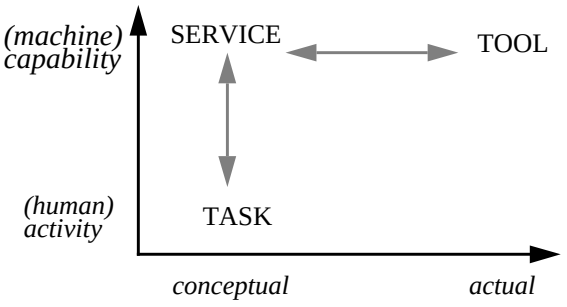

Figure 2: Relationships of Tools, Tasks, and Services.

instance, one might consider that the environment provides some capability (e.g., an environment's testing service); or one might consider that a human user performs some task using the environment (e.g., the human activity of testing). Whichever view one takes, both refer to the same notion, (e.g., a human using a piece of software to test the output of an engineering process).

In brief, services are abstract capabilities of the environment, tasks make use of and provide context for those capabilities, and tools are the actual executable software components that realize environment services. Figure 2 illustrates the distinction between these concepts. *Service* can be contrasted with *Tool* along an axis of Conceptual vs. Actual, or it can be contrasted with *Task* along an axis of Capability vs. Activity.

The PSE reference model is a catalog of service descriptions spanning the functionality of a populated environment. The service descriptions are grouped by several different categories, including degrees of abstraction, granularity, or functionality. The highest-level division classifies services either as end-user or framework services. The former includes services that directly support the execution of a project (i.e., services that tend to be used by those who directly participate in the execution of a project such as engineers, managers, and secretaries). The latter services either pertain to users who facilitate, maintain, or improve the operation of the computer system itself (e.g., a human user performing such tasks as tool installation) or are used directly by other services in the environment. End-user services are further subdivided into Technical Engineering, Technical Management, Project Management, and Support services. The first three of these groups partition the execution of a project into engineering, management, and a middle category that partakes of both. The fourth group, Support services, is orthogonal to the other three, since it includes capabilities potentially used by all other users, such as word processing, mail, and publication.

Figure 3 illustrates the logical relationships between these service groups. Framework services form a central core with a potential relationship to all other services in the environment. Support services underlie the other end-user services. The remaining three groups, Technical Engineering, Technical Management, and Project Management, surround the Framework services and make use of the Support services. In addition, services from these three groups may have relationships with each other. It is not the intention that the boundaries of the parts of this drawing explicitly indicate interfaces, since this figure is drawn at the level of service *groups*, not of individual services. Thus, it must be stressed that while a drawing such as this attempts to suggest in a very general manner how the high-level service groups logically relate to each other, there is an express intention to avoid any sort of architectural implication. The reference

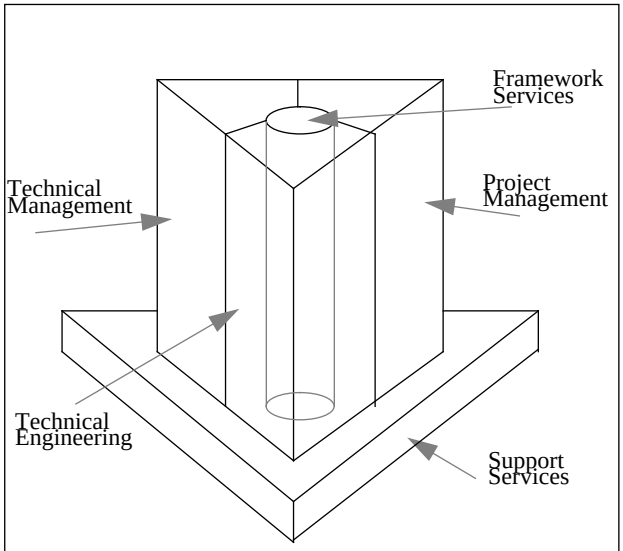

Figure 3: A Graphical Depiction of the Reference Model Service Groups.

model is a conceptual, not an actual, model, and no architectural choices are intended by this figure. To emphasize this point the same set of service groups, with the same set of potential relationships, could also be illustrated by Figure 4.

The key point is that the figures are illustrative only and do not in any way connote layering of services, architectural decisions, or implementation choices for an actual environment.

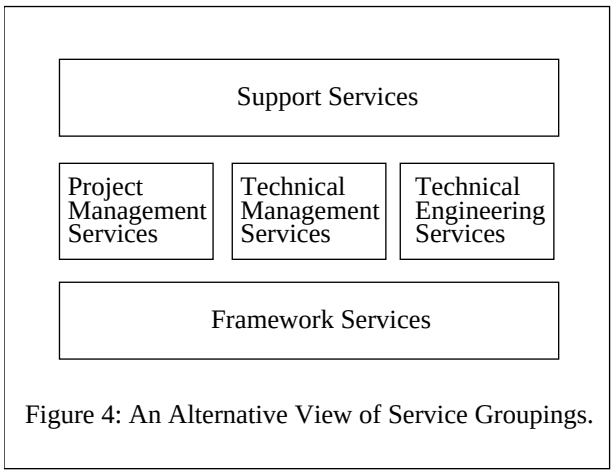

Figure 4: An Alternative View of Service Groupings.

3. DESCRIPTION OF THE REFERENCE MODEL SEVICES

The reference model distinguishes two groups of services: end-user services and framework services. In this section we briefly review the services that have been defined in each of these groups, and examine the distinction between host and target system support in a PSE. Full details of the reference model services can be found in the complete reference model document [PSESWG 92].

3.1. End-User Services

Each of the end-user service categories (Technical Engineering, Technical Management, Project Management, and Support services) is further subdivided by engineering domain, by user role, or life-cycle phase. Technical Engineering services focus on the technical aspects of project development. These services support activities related to the specification, design, implementation, and maintenance of systems. They are subdivided by specific engineering domains (e.g., Software Engineering). In addition to 'traditional' engineering domains, the reference model also considers life-cycle processes to be an area for which an engineering discipline is appropriate, and services related to that domain are included here as well. Within an engineering domain the processes used in the life cycle of a project define a series of tasks, each requiring services for its support. Thus, within the software engineering domain, tasks typically include designing and coding, which require services like compilation and testing. The following Technical Engineering services are defined in the reference model:

System Engineering Services:
- System Requirements Engineering
- System Design and Allocation
- System Simulation and Modeling
- System Static Analysis
- System Testing
- System Integration
- System Re-engineering
- Host-Target Connection
- Target Monitoring
- Traceability

Software Engineering Services:
- Software Requirements Analysis
- Software Design
- Software Simulation and Modeling
- Software Verification
- Software Generation
- Compilation
- Software Static Analysis
- Debugging
- Software Testing
- Software Build
- Software Reverse Engineering
- Software Re-engineering
- Software Traceability

Life-Cycle Process Engineering Services:
- Process Definition
- Process Library
- Process Exchange
- Process Usage

Technical Management provides services that are closely related to engineering activities. However, these services pertain to activities that are often equally shared by engineers and managers; the operations of these services do not clearly fall into one or the other category, but fall into a middle category that partakes of both Technical Engineering and Project Management. These services provide a managerial complement to engineering activities in the areas of Configuration Management, Reuse, and Metrics.

The following Technical Management services are defined in the reference model:
- Configuration Management
- Change Management
- Reuse Management
- Metrics

Project Management services are relevant to the overall success of the enterprise. These services support activities related to developing and executing a project, including such things as scheduling, planning, and tracking the

project's overall progress. These activities span the lifetime of a project from inception through deployment and maintenance.

The reference model describes the following Project Management services:

- Scheduling
- Estimation
- Risk Analysis
- Tracking

Support services focus on tasks and activities common to all users of a PSE, regardless of the domain, role, or lifecycle phase in which the activity is taking place. Support services are needed by virtually all users of the computer system. They include services associated with processing, formatting, and disseminating human-readable data, including several common text and figure processing services, as well more specialized publishing, user communication, and presentation services. They also include administration services that provide support for use of the PSE itself.

The reference model describes the following Support Services:

- Common Support Services:
  - Text Processing
  - Numeric Processing
  - Figure Processing
  - Audio and Video Processing
  - Calendar and Reminder
  - Annotation
- Publishing Services
- Presentation Preparation Services
- User Communication Services:
  - Mail
  - Bulletin Board
  - Conferencing
- PSE Administration Services:
  - Framework Administration and Configuration
  - Tool Installation and Customization
  - PSE User and Role Management
  - PSE Resource Management
  - PSE Status Monitoring
  - PSE Diagnostic
  - PSE Interchange
  - PSE User Access

3.2. Framework Services

These services comprise the infrastructure of a PSE. They include those services that jointly provide support for applications, CASE tools, etc. and that are commonly referred to as "the environment framework." Since 1989, the National Institute of Standards and Technology (NIST) has sponsored a series of workshops developing a reference model for environment frameworks. The product of that group is a document published jointly by NIST and the European Computer Manufacturers' Association (ECMA) that is commonly known as the "NIST/ECMA Frameworks Reference Model" [NIST RM]. This document contains detailed descriptions of fifty framework services. The PSESWG elected essentially to use the NIST document in its entirety, and the PSESWG reference model simply contains abstracts of the more extensive descriptions found in the NIST/ECMA document.

In addition to the NIST/ECMA set of framework services, the PSESWG has also chosen to include some other infrastructure services not present in the NIST/ECMA document. The PSESWG has abstracted several services from the work of the "Draft Guide to the POSIX Open Systems Environment" sponsored by the IEEE, known as "POSIX.0," as a source for these [POSIX.0].

In both cases, the PSESWG reference model has abstracted the service descriptions. A reader of the PSESWG reference model is urged to consult these other two documents for a full description of the infrastructure services. It should also be noted that, at the infrastructure level, some services are actually groups of services which in turn contain lower-level services.

The reference model defines the following framework services:

- Operating System
- Object Management
- Policy Enforcement
- Process Management
- Communication
- User Interface
- User Command Interface
- Network

In addition to the five groups referenced here, the NIST/ECMA Frameworks reference model contains services related to framework administration and configuration; these are included in the PSE Administration services section of the PSE reference model.

3.3. Place Of The Target System In The Model

While the target system may be the same as the development system, there is no requirement that this be so. The PSE reference model therefore differentiates between the services available on the host machine used in the development of computer-based systems and services on the target machine upon which the developed system will execute. Within the NGCR program some of the details of target system functionality are described elsewhere. One source for these details is the "Operating Systems Standards Working Group Reference Model," June, 1990 [OSSWG RM]. Other services, in particular those relating to connection and mon-

itoring of target system services to the development system, are part of this PSEWG reference model, and are included in the End-User Services listed in Section 3.1.

4. CONCLUSIONS AND FUTURE WORK

In this paper we have reviewed the main elements of a PSE reference model that we have defined as a necessary step toward the goal of selecting standards that will facilitate the assembly of a PSE from COTS products. Producing such a reference model has been a major undertaking involving a great deal of resources. We believe, however, that this effort has been very beneficial to our PSESWG goals in a number of ways:

- it is a focal point for producing a common set of concepts, terminology, and issues that are an essential basis for making progress in a large, multi-organizational effort such as the PSESWG.
- it is a framework for categorizing existing and proposed standards and products as a necessary precursor to standards selection.
- it is a public document that illustrates our intentions to the PSE research community, attracting people to attend, comment, and contribute to our efforts.

Additionally, we also believe that our work has much wider implications for others working in the PSE area, and in the software engineering area in general:

- it is an example of the kind of reference material that must be developed in the area of PSEs to provide a deeper understanding of a number of the issues that need to be addressed.
- it is a usable, practical document that can help in the analysis and evaluation of complementary and competing PSE standards and products — a task in which many organizations require help and support.
- it is a demonstration of the effectiveness of leveraging the talents and experience of government, academia, and industry to produce useful results that are of benefit to each of these communities.

Looking to the future, the release date for Version 1.0 of the reference model was February 1993[7] after which the document will be revised periodically. To aid the work by members of PSESWG, the reference model has been reviewed by members of other working groups, notably the NIST ISEE workshops, the Technical Group on Reference Models (TGRM) of the ECMA Technical Committee 33 (TC33), and several of the contributing experts of the inter- national Portable Common Interface Set (PCIS) program. These persons have made many valuable contributions toward the final document. In addition to developing the reference model, PSESWG also supports other related activities. For example, the reference model is being used by PSESWG members in several mapping activities, making use of the reference model as a basis for examining actual environments. More are planned during the coming year, and a future document will detail the results of these mapping activities. Additionally, a catalog of available technology has been compiled and will periodically be updated.

PSESWG has now moved into a second stage, which is to examine actual standards and products selected from the catalog of available technology. Two teams of working group members have been formed, one of which is investigating standards and products related to framework services, and the other examining standards and products related to data interfaces. These two groups will examine as many of these items as is feasible. The result of these examinations will be formal characterizations of the important interfaces, as well as a list of candidate standards for these interfaces. PSESWG's final activity will be to make actual selections of interface standards which will then be collectively listed in a single NGCR PSE standard. Accompanying such a list will be a document describing detailed considerations of the relationships, overlaps, omissions, and options that must be considered in using the collection of standards.

Finally, there will likely be other merits of the reference model in addition to its planned use by NGCR. Its use as a basis for a common set of concepts and terminology with which to discuss the PSE domain will be a very useful contribution to the whole PSE community. Similarly, the reference model has potential value in the study and analysis of tool integration and may help in characterizing tool capabilities.

As NGCR funding for the PSESWG effort will be cut after September 1993, PSESWG's final activity will be to document the work that has been carried out and the progress that has been made in a detailed closing report. In addition to information on the results of using this reference model to examine various interface standards and products, that report will include information on the standards examined, how a follow-on group might proceed to complete the original PSESWG objective, and advice to program managers on how to make selections of standards and products for their own projects in the near term.

7. The reference model documents are available from any of the authors and in electronic form from the PSESWG archive. Electronic mail inquiries should be sent to "psearch@nadc.navy.mil" with a subject line of "help".

Acknowledgments

A large number of people have contributed to the work of the NGCR PSESWG in general and to the production of the PSE reference model in particular. We gratefully acknowledge their contribution to the work reported in this paper.

We also thank the internal SEI referees of this paper for their very valuable comments.

An extended version of this paper will appear in C*omputer Standards and Interfaces* Journal.

The SEI is sponsored by the U.S. Department of Defense.

5. REFERENCES

[Brown/Feiler 92] *An Analytical Technique for Examining Integration in a Project Support Environment*, Fifth Symposium on Software Development Environments, ACM, December 1992.

[NIST RM] *Reference Model for Frameworks of Software Engineering Environments.* National Institute for Standards and Technology, Special Publication Number 500-201, December 1991.

[OSSWG RM] *Reference Model for Embedded Operating Systems.* NGCR Operating System Standards Working Group, June 1990.

[POSIX.0] *Draft Guide to the POSIX Open Systems Environment.* P1003.0, IEEE, June 1992.

[PSESWG 93] *NGCR Reference Model for Project Support Environments*, NGCR Project Support Environments Working Group, Version 1.0, NAWC/AD, Warminster, PA, February 1993.

[Stoneman 80] J.N. Buxton, “Requirements for Ada Programming Support Environments — Stoneman”, U.S. Department of Defense, February 1980.